Установка linux mint 13 на raid1

Введение

Уже давно присматривался к данной ОС в качестве основной на десктопе, но все как то не было времени, а возможно и желания, сделать переход с Windows. Но все таки этот момент настал, да и срок окончанию поддержки Windows XP, которую я использую в качестве основной, заканчивается через год. Так что это хороший повод перейти на Linux, теперь уже и на рабочем месте, а не только на серверах. Но в процессе установки столкнулся с некоторыми особенностями.

Подготовка к установке

Под рукой оказался хороший компьютер, а именно наличие 2x SSD - Intel 520 Series. Которые я хотел объединить в программный raid 1 и на него установить систему. Как всегда, записал образ linux mint 13 (mate) на диск и загрузил live cd. После чего запустил установку системы, но увидел, что из установщика самого Mint создать программный raid 1 к сожалению нельзя, но ведь очень хочется.

Можно конечно поставить систему на один диск, и уже потом создать raid1, но данный способ предполагает много операций и действия, а так как мы, системные администраторы, народ ленивый, то пойдем по другому пути.

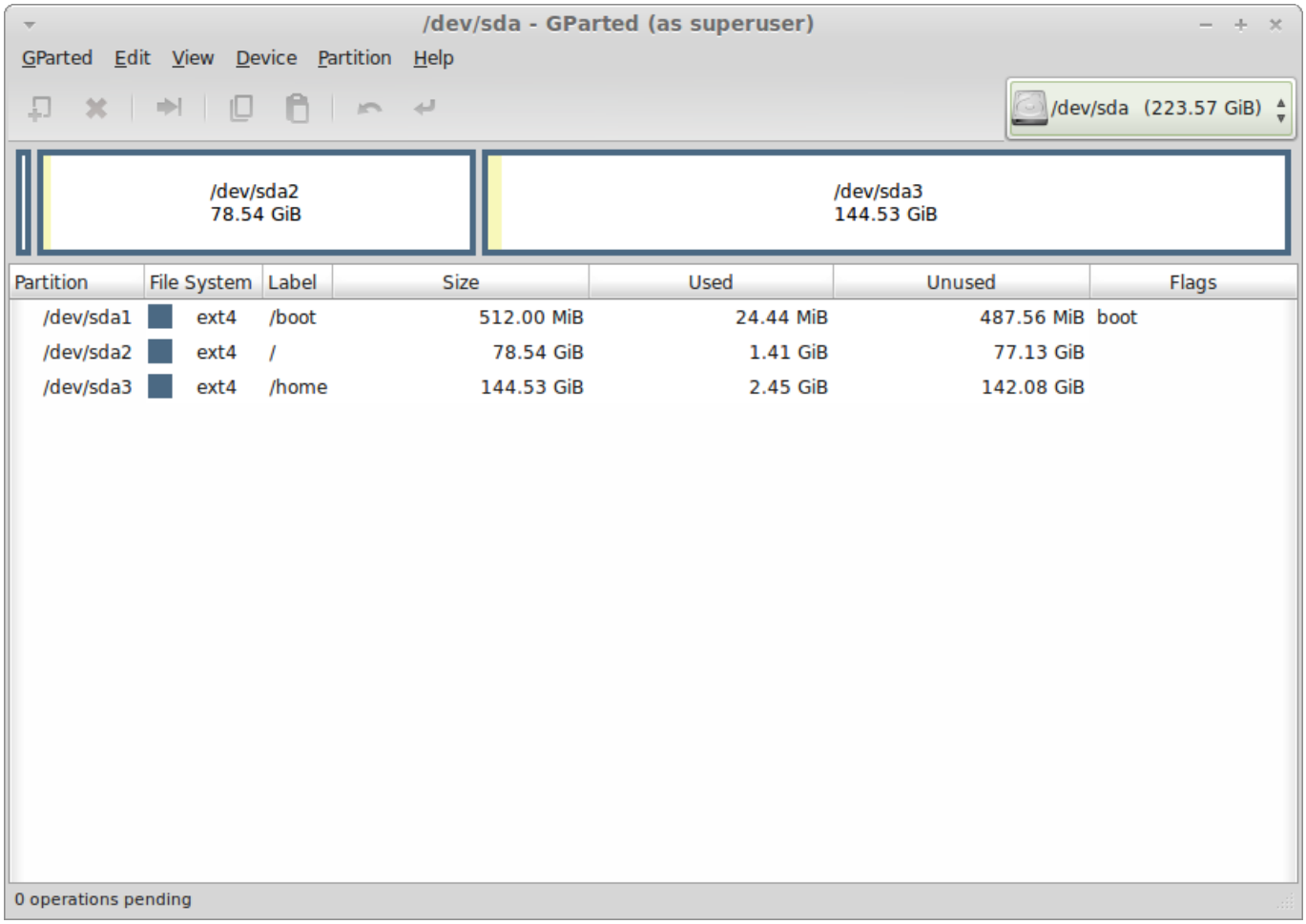
Итак, у нас в наличии следующие диски

```
# fdisk -l /dev/sda

Disk /dev/sda: 240.1 GB, 240057409536 bytes
255 heads, 63 sectors/track, 29185 cylinders, total 468862128 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/sda doesn't contain a valid partition table

# fdisk -l /dev/sdb

Disk /dev/sdb: 240.1 GB, 240057409536 bytes
255 heads, 63 sectors/track, 29185 cylinders, total 468862128 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/sdb doesn't contain a valid partition table
```

Диски решил разбить следующим образом, так как памяти на компьютере 16 Gb, то swap раздел вообще решил не создавать. В общем то, как разбить диск дело индивидуальное и зависит от конкретных задач и условий.

```
/boot - 512 Mb
/ - 80 Gb
/home - все остальное
```

На всех партициях файловая система ext4. В результате у меня получилось примерно следующая картина



Создание raid массива

Устанавливаем mdadm, так как по умолчанию он не установлен

```
# aptitude install mdadm
```

И запускаем создание самих разделов

```
# sudo mdadm --create /dev/md0 --level=1 --raid-devices=2 /dev/sd[ab]1
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md0 started.
```

```
# sudo mdadm --create /dev/md1 --level=1 --raid-devices=2 /dev/sd[ab]2
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md0 started.

# sudo mdadm --create /dev/md2 --level=1 --raid-devices=2 /dev/sd[ab]3
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md0 started.
```

Наблюдаем за состоянием процесса создания массива

```
# cat /proc/mdstat
Personalities : [raid1]
md2 : active raid1 sdb3[1] sda3[0]
      151549880 blocks super 1.2 [2/2] [UU]
        resync=DELAYED

md1 : active raid1 sdb2[1] sda2[0]
      82353080 blocks super 1.2 [2/2] [UU]
      [==>..................]  resync = 10.6% (8795776/82353080)
finish=6.0min speed=203489K/sec

md0 : active raid1 sdb1[1] sda1[0]
      524276 blocks super 1.2 [2/2] [UU]

unused devices: <none>
```

Так как у нас в наличие SSD, то можно ускорить процесс ребилда массива

```
# echo 500000 > /proc/sys/dev/raid/speed_limit_max
```

Как видим скорость возросла прилично

```
# cat /proc/mdstat
Personalities : [raid1]
md2 : active raid1 sdb3[1] sda3[0]
      151549880 blocks super 1.2 [2/2] [UU]
      [======>.............]  resync = 33.0% (50021568/151549880)
finish=3.3min speed=500834K/sec

md1 : active raid1 sdb2[1] sda2[0]
      82353080 blocks super 1.2 [2/2] [UU]

md0 : active raid1 sdb1[1] sda1[0]
      524276 blocks super 1.2 [2/2] [UU]

unused devices: <none>
```

После полной синхронизации всех дисков, создаем фс

```
# mkfs.ext4 /dev/md0
```

```
mke2fs 1.42 (29-Nov-2011)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
Stride=0 blocks, Stripe width=0 blocks
131072 inodes, 524276 blocks
26213 blocks (5.00%) reserved for the super user
First data block=1
Maximum filesystem blocks=67633152
64 block groups
8192 blocks per group, 8192 fragments per group
2048 inodes per group
Superblock backups stored on blocks:
        8193, 24577, 40961, 57345, 73729, 204801, 221185, 401409

Allocating group tables: done
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done

# mkfs.ext4 /dev/md1
mke2fs 1.42 (29-Nov-2011)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
5152768 inodes, 20588270 blocks
1029413 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=4294967296
629 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632,
2654208,
        4096000, 7962624, 11239424, 20480000

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

# mkfs.ext4 /dev/md2
mke2fs 1.42 (29-Nov-2011)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
```

```
Stride=0 blocks, Stripe width=0 blocks
9478144 inodes, 37887470 blocks
1894373 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=4294967296
1157 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632,
2654208,
        4096000, 7962624, 11239424, 20480000, 23887872

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
```

Подготовка к chroot

Создаем две папки, **target** - здесь будет располагаться наша будущая система, **source** - здесь будет располагаться сами бинарные файлы с live cd диска.

```
# mkdir /source /target /target/boot
```

А теперь монтируем наши будущие диски в соответствующие папки. Напомню, что на /dev/md1 у нас располагается корень системы, а на /dev/md0 соответственно /boot

```
# mount /dev/md1 /target/
# mount /dev/md0 /target/boot
```

После этого монтируем live-cd и копируем все данные с диска на наши md устройства (на моих дисках Intel SSD заняло ~7 минут)

```
# mount -o loop -t squashfs /cdrom/casper/filesystem.squashfs /source/
# rsync -avz /source/ /target/
```

Копируем кеш apt и прописываем dns сервер

```
# cp /var/cache/apt/archives/* /target/var/cache/apt/archives/
cp: omitting directory `/var/cache/apt/archives/partial'

# echo "nameserver 8.8.8.8" > /target/etc/resolv.conf
```

Ну и собственно производим монтирование необходимых точек и файловых систем, для перехода в chroot

Теперь подготавливаем нашу систему для chroot

```
# mount --bind /dev/ /target/dev/
# mount --bind /dev/shm/ /target/dev/shm/
# mount --bind /dev/pts/ /target/dev/pts/
# mount --bind /proc/ /target/proc/
# mount --bind /sys/ /target/sys/
# mount --bind /tmp/ /target/tmp/
```

Теперь у нас все готово для выполнения chroot

Установка grub2

Собственно делаем сам chroot

```
# chroot /target/
```

Снова устанавливаем mdadm и mc (по желанию)

```
# aptitude install mc mdadm
The following NEW packages will be installed:
  mc mc-data{a} mdadm postfix{a}
0 packages upgraded, 4 newly installed, 0 to remove and 103 not upgraded.
Need to get 0 B/3,774 kB of archives. After unpacking 10.3 MB will be used.
Do you want to continue? [Y/n/?] y
```

В процессе установки вы увидите сообщения вида

```
Processing triggers for menu ...
Processing triggers for initramfs-tools ...
update-initramfs: Generating /boot/initrd.img-3.2.0-23-generic
cryptsetup: WARNING: could not determine root device from /etc/fstab
Warning: No support for locale: en_US.utf8
mdadm: cannot open /dev/md/0: No such file or directory
mdadm: cannot open /dev/md/1: No such file or directory
mdadm: cannot open /dev/md/2: No such file or directory
Processing triggers for libc-bin ...
ldconfig deferred processing now taking place
```

Это вполне нормально, так просто игнорируем их.

Устанавливаем grub

```
# dpkg-reconfigure grub-pc
Replacing config file /etc/default/grub with new version

# grub-install /dev/sda
Installation finished. No error reported.

# grub-install /dev/sdb
```

```
Installation finished. No error reported.
```

Генерируем grub.cfg

```
# update-grub
Generating grub.cfg ...
Found linux image: /boot/vmlinuz-3.2.0-23-generic
Found initrd image: /boot/initrd.img-3.2.0-23-generic
Found memtest86+ image: /memtest86+.bin
done
```

Заключительные штрихи

Меняем пароль на пользователя root и создаем не привилегированного пользователя

```
# passwd root
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully

# useradd alex
# passwd alex
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully

# usermod -G sudo alex
# id alex
uid=1000(alex) gid=1000(alex) groups=1000(alex),27(sudo)
```

Все, что нам осталось сделать, это прописать данные в fstab. Что мы собственно и делаем. Для начала определяем UUID наших дисков, которые мы и будем использовать в fstab вместо /dev/sd[ab]

```
# blkid /dev/md0
/dev/md0: UUID="e2f24dfa-9428-4908-a819-bec5dd39e72c" TYPE="ext4"

# blkid /dev/md1
/dev/md1: UUID="15d20518-34b2-414a-8f5d-33006fcf6105" TYPE="ext4"

# blkid /dev/md2
/dev/md2: UUID="ea68e109-1d50-4750-b76e-51e497407d10" TYPE="ext4"
```

В результате у нас должен получится следующий файл

```
# cat /etc/fstab
tmpfs           /dev/shm                tmpfs   defaults        0 0
devpts          /dev/pts                devpts  gid=5,mode=620  0 0
sysfs           /sys                    sysfs   defaults        0 0
```

```
proc            /proc                   proc    defaults        0 0

# /dev/md0
UUID="e2f24dfa-9428-4908-a819-bec5dd39e72c" /boot       ext4    defaults
1 2

# /dev/md1
UUID="15d20518-34b2-414a-8f5d-33006fcf6105" /           ext4    defaults
1 1

# /dev/md2
UUID="ea68e109-1d50-4750-b76e-51e497407d10" /home       ext4    defaults
1 2
```

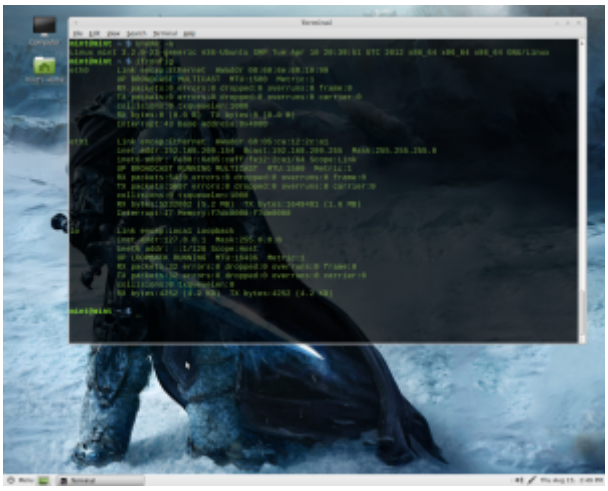
Вот собственно и все. Выходим из chroot, перегружаем систему и пользуемся Linux Mint :)

```
# exit
# reboot
```

После загрузки мы получим такую систему

```
# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/md1         79G  4.4G   71G   6% /
udev             12G  4.0K   12G   1% /dev
tmpfs           4.7G  1.1M  4.7G   1% /run
none            5.0M     0  5.0M   0% /run/lock
none             12G   76K   12G   1% /run/shm
/dev/md0        509M   54M  430M  12% /boot
/dev/md2        145G  2.3G  135G   2% /home
```

Как то так стал выглядеть мой рабочий стол



Заключение

Вот такими не хитрыми действиями мы смогли установить Linux Mint на программный raid.

Честно говоря, я не понял, почему разработчики не добавили возможность создания программного raid массива в самом инсталяторе. Возможно из-за того, что дистрибутив рассчитан прежде всего на простых пользователей, для которых raid просто непонятное слово? :) Сложно сказать, оставим это на их совести